Mistral 7B

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, William El Sayed

ABSTRACT

We introduce Mistral 7B, a 7-billion-parameter language model engineered for superior performance and efficiency. Mistral 7B outperforms the best open 13B model (Llama 2) across all evaluated benchmarks, and the best released 34B model (Llama 1) in reasoning, mathematics, and code generation. Our model leverages grouped-query attention (GQA) for faster inference, coupled with sliding window attention (SWA) to effectively handle sequences of arbitrary length with a reduced inference cost. We also provide a model fine-tuned to follow instructions, Mistral 7B – Instruct, that surpasses the Llama 2 13B – chat model both on human and automated benchmarks. Our models are released under the Apache 2.0 licence.

Code: https://github.com/mistralai/mistral-src Webpage: https://mistral.ai/news/announcing-mistral-7b/

1 Introduction

In the rapidly evolving domain of Natural Language Processing (NLP), the pursuit of higher model performance often necessitates an increase in model size. However, this scaling tends to raise computational costs and inference latency, thereby creating barriers to deployment in practical, real-world scenarios. In this context, the search for balanced models that deliver both high-level performance and efficiency becomes critically important. Our model, Mistral 7B, demonstrates that a carefully designed language model can achieve high performance whilst maintaining efficient inference. Mistral 7B outperforms the previous best 13B model (Llama 2, *[26]*) across all tested benchmarks, and surpasses the best 34B model (LLaMa 34B, *[25]*) in mathematics and code generation. Furthermore, Mistral 7B approaches the coding performance of Code-Llama 7B *[20]*, without sacrificing performance on non-code-related benchmarks.

Mistral 7B leverages grouped-query attention (GQA) *[1]*, and sliding window attention (SWA) *[6, 3]*. GQA significantly accelerates inference speed, and also reduces the memory requirement during decoding, allowing for higher batch sizes and thus higher throughput, a crucial factor for real-time applications. Additionally, SWA is designed to handle longer sequences more effectively at a reduced computational cost, thereby alleviating a common limitation in LLMs. These attention mechanisms collectively contribute to the enhanced performance and efficiency of Mistral 7B.

Mistral 7B is released under the Apache 2.0 licence. This release is accompanied by a reference implementation facilitating easy deployment either locally or on cloud platforms such as AWS, GCP, or Azure using the vLLM [17] inference server and SkyPilot. Integration with Hugging Face is also streamlined for easier adoption. Moreover, Mistral 7B is designed for ease of fine-tuning across a wide range of tasks. As a demonstration of its adaptability and superior performance, we present a chat model fine-tuned from Mistral 7B that significantly outperforms the Llama 2 13B - Chat model.

Mistral 7B represents a significant step in balancing the objectives of achieving high performance whilst maintaining the efficiency of large language models. Through our work, our aim is to help the community create more affordable, efficient, and high-performing language models that can be utilised in a broad spectrum of real-world applications.

2 Architectural details

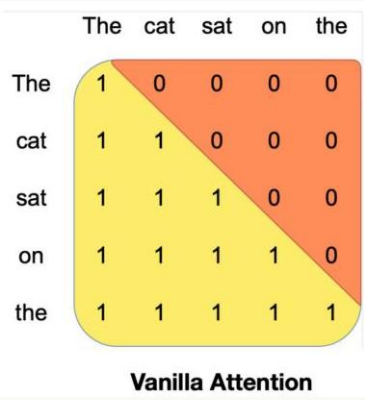
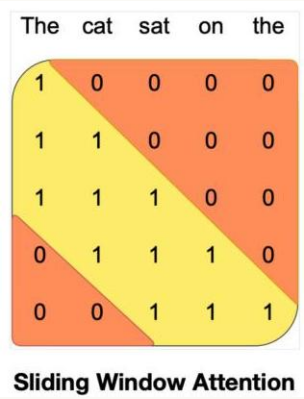
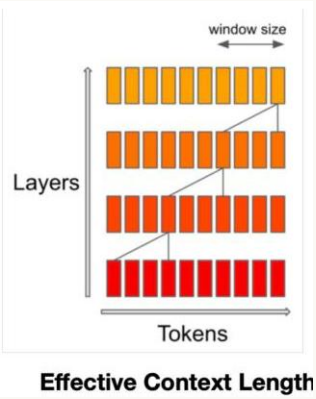


Figure 1: Sliding Window Attention. The number of operations in vanilla attention is quadratic in the sequence length, and the memory increases linearly with the number of tokens. At inference time, this incurs higher latency and smaller throughput due to reduced cache availability. To mitigate this issue, we use sliding window attention: each token can attend to at most W tokens from the previous layer (here, $W = 3$). Note that tokens outside the sliding window still influence next word prediction. At each attention layer, information can move forward by W tokens. Hence, after k attention layers, information can move forward by up to $k \times W$ tokens.





Mistral 7B is based on a transformer architecture [27]. The main parameters of the architecture are summarised in Table 1. Compared to Llama, it introduces a few changes that we summarise below.

Sliding Window Attention. SWA exploits the stacked layers of a transformer to attend information beyond the window size W. The hidden state in position i of the layer k, h_i, attends to all hidden states from the previous layer with positions between $i - W$ and i. Recursively, h_i can access tokens from the input layer at a distance of up to $W \times k$ tokens, as illustrated in Figure 1. At the last layer, using a window size of $W = 4096$, we have a theoretical

attention span of approximately 131K tokens. In practice, for a sequence length of 16K and $W = 4096$, modifications made to FlashAttention [11] and xFormers [18] yield a 2x speed improvement over a vanilla attention baseline.

Parameter	Value
dim	4096
n_layers	32
head_dim	128
hidden_dim	14336
n_heads	32
n_kv_heads	8
window_size	4096
context_len	8192
vocab_size	32000

Table 1: Model architecture.

Rolling Buffer Cache. A fixed attention span means that we can limit our cache size using a rolling buffer cache. The cache has a fixed size of W, and the keys and values for the timestep i are stored in position $i \bmod W$ of the cache. As a result, when the position i is larger than W, past values in the cache are overwritten, and the size of the cache stops increasing. We provide an illustration in Figure 2 for $W = 3$. On a sequence length of 32k tokens, this reduces the cache memory usage by 8x, without impacting the model quality.